An illustration of new methods in machine condition monitoring, Part I: Stochastic resonance

IOP Conf. Series: Journal of Physics: Conf. Series **842** (2017) 012058 doi:10.1088/1742-6596/842/1/012058

An illustration of new methods in machine condition monitoring, Part I: stochastic resonance

K. Worden[1], I. Antoniadou[1], S. Marchesiello[2], C. Mba[2] and L. Garibaldi[2]

[1]Dynamics Research Group, Department of Mechanical Engineering, University of Sheffield, Mappin Street, Sheffield S1 3JD, UK.
[2]Department of Mechanical and Aerospace Engineering, Politecnico di Torino, 10129, Torino, Italy.

E-mail: k.worden@sheffield.ac.uk

Abstract. There have been many recent developments in the application of data-based methods to machine condition monitoring. A powerful methodology based on machine learning has emerged, where diagnostics are based on a two-step procedure: extraction of damage-sensitive features, followed by unsupervised learning (novelty detection) or supervised learning (classification). The objective of the current pair of papers is simply to illustrate one state-of-the-art procedure for each step, using synthetic data representative of reality in terms of size and complexity. The first paper in the pair will deal with feature extraction.

Although some papers have appeared in the recent past considering stochastic resonance as a means of amplifying damage information in signals, they have largely relied on *ad hoc* specifications of the resonator used. In contrast, the current paper will adopt a principled optimisation-based approach to the resonator design. The paper will also show that a discrete dynamical system can provide all the benefits of a continuous system, but also provide a considerable speed-up in terms of simulation time in order to facilitate the optimisation approach.

1. Introduction

The aim of machine condition monitoring is to detect incipient damage in (usually rotating) machinery before it grows to the point where a failure of the machine occurs. This entails detecting the signature of small damage in a component from signals which may be strongly corrupted by the presence of noise from other elements of the machine or environment. When the components of interests are gears or bearings, the 'signature' of damage in vibration data can take the form of a train of impulses. In a gearbox, a crack in a tooth may cause increased flexibility of that tooth, which can manifest as an impulse when the tooth comes into mesh; the period of the impulse train will then be associated with the meshing frequency. In a bearing, a crack or spall in a race will cause an impulse when a ball runs over the damage; the period of the impulse train is then associated with one of the bearing characteristic fault frequencies. There is clearly some benefit in being able to detect the presence of impulses in a noisy background; establishing the period of the impulse train can then give information about which gear in a system, or which race in a bearing, is damaged.

One approach to the problem described above might be to selectively increase the effect of the impulse in the signal relative to the noise background, and this is where the phenomenon of

stochastic resonance becomes of interest. The phenomenon was first demonstrated in [1] in the context of climatology, although applications in various fields followed [2]. In the context of the problem here, stochastic resonance can be used as a signal processing tool to increase the signal-to-noise ratio (SNR) of a mixture of deterministic signal and noise. This is accomplished by passing the noisy signal through a nonlinear system; for carefully tuned nonlinear systems, the response can show amplification of the deterministic component relative to the noise. Although the technique is most often applied in order to amplify a small harmonic signal in noise, it can also be used for the impulse trains of the condition monitoring context [3, 4, 5]. In a previous article by some of the authors [6], the approach was illustrated on a number of condition monitoring data sets and was shown to be effective, if the parameters of the nonlinear oscillator could be tuned appropriately. Although the results were very good, it was found that parameter tuning required care and expertise on the part of the analysts. The objective of the current paper is to introduce an *optimisation* based tuning scheme, independent of the user, and to validate it via application to two synthetic data sets. (While this paper was in preparation, the authors became aware of the reference [7], which also explores some of the concepts discussed here.)

The layout of the paper is as follows: the following section briefly explains how stochastic resonance can amplify the effect of an impulse train in noise, and how the oscillator parameters might be tuned using an optimisation scheme. Section Three outlines the background theory for the optimisation scheme chosen for this paper - *Self Adaptive Differential Evolution* (SADE). Section Four describes how the synthetic data for the case studies were constructed, and results for the data are presented in Section Five. The paper ends with brief conclusions.

2. Stochastic Resonance

Rather than describe stochastic resonance in general, the discussion here will motivate the approach via the explicit nonlinear oscillator considered in this paper, which, in its simplest form, is a first-order cubic oscillator of the form,

$$\dot{y} - ay + by^3 = p(t) \tag{1}$$

which is a system with a cubic internal restoring force, driven by a mixture of a single impulse component and noise. In the context of stochastic resonance, the amplitude of the impulsive component will usually be small compared to the noise. Now, the potential $V(y)$ associated with the internal force is,

$$V(y) = -\frac{a}{2}y^2 + \frac{b}{4}y^4 \tag{2}$$

which is for $a > 0$ and $b > 0$, a twin-well oscillator as illustrated in Figure 1 (for the values $a = 40$, $b = 1$).

Now, consider a particle moving in this potential. If the level of forcing is low, the particle will remain very close to one of the stable equilibria at $\pm\sqrt{2a/b}$. Without loss of generality, consider the particle to be in the negative well. Suppose that the noise signal is a zero-skewness process; the mixture of the impulse and noise will have a small degree of skewness because of the impulse, however this will usually be imperceptible because A_0 is small compared to X. If one 'turns up' the level of forcing the particle will begin to move between the limits given by $V(y)$, but corresponding to a higher value of y. At the limit where the particle has maximum amplitude, yet remains within the well, the oscillations will be skewed - the signal will be 'drawn out' at the side towards the origin. This 'drawing out' will amplify the skewing effect of the impulse, those indicating the presence of the impulse. By tuning the parameters a and b, one is able to amplify the skewing effect and increase detectability of the impulse; however, one must take care not to choose parameters which allow the particle to pass into the positive well, or the effect will be lost.

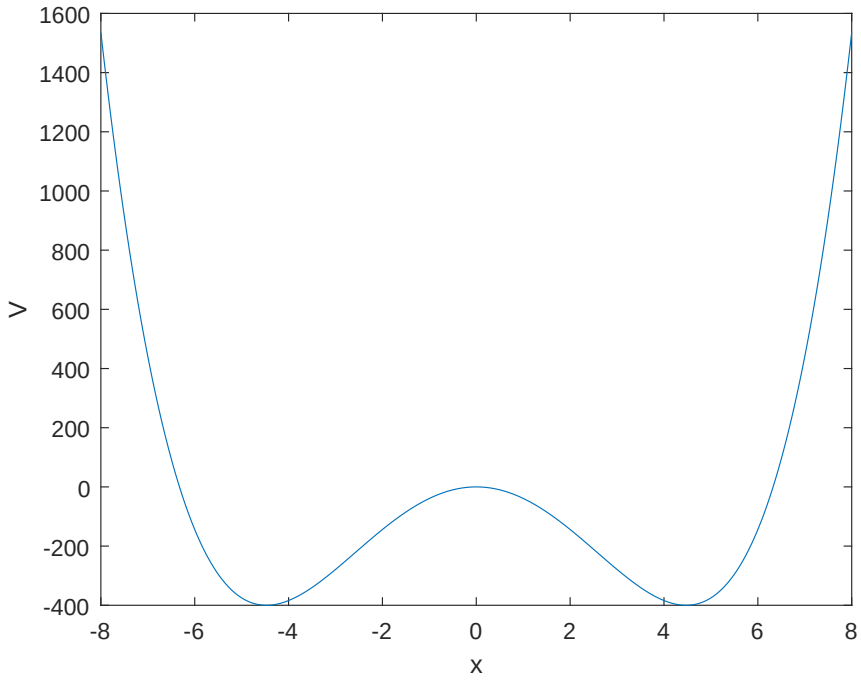

Figure 1. Twin-well potential for the oscillator in equation (1) with $a = 40$ and $b = 1$.

A previous paper by some of the authors here [6], carried out a careful tuning of the parameters in order to demonstrate how effectively impulses could be detected in a noise background. It was clear from the paper that an automated tuning procedure, requiring less intervention from the user, was desirable. The current study will show that this is possible using an optimisation framework. The idea will be to optimise the parameters according to some measure of the 'drawing out' effect, while constraining the particle to a single well. The drawing out or asymmetry could in principle be optimised by maximising the skewness of the oscillator response; however, in [6], it was preferred to use the kurtosis. This approach makes sense because the kurtosis of a signal is a measure of its 'spikiness', and this is increased by emphasising the impulses. A simple penalty term can be used to prevent the particle crossing the origin, or to rule out candidate parameters which make the oscillator unstable[1]. So, the objective function for optimisation involves forcing the nonlinear oscillator with the signal of interest and then computing the kurtosis of the response. Unfortunately, this is time-consuming because it requires solving an initial value problem for each evaluation of the objective function. The solution to this problem is quite simple; rather than using a continuous-time oscillator, one passes to a discrete-time variant. For this case, forward predictions using the oscillator model will be much faster.

Starting from the oscillator model of equation (1), one can approximate the derivative using the discrete difference $(y_{i+1} - y_i)/\Delta t$, yielding,

$$y_{i+1} = (1 + a\Delta t)y_i - b\Delta t y_i^3 + \Delta t p(t) \tag{3}$$

[1] In fact, the kurtosis of the response signal will be increased if the particle occasionally crosses between wells; however, if this happens too often, the desired effect is lost, especially as jumps can occur that are not caused by impulses. For this reason, the well-crossing constraint is applied here.

Incorporating the rescaling ratio $R = \Delta t_r/\Delta t$, discussed in [6], then gives,

$$y_{i+1} = (1 + \frac{a\Delta t_r}{R})y_i - \frac{b\Delta t}{R}y_i^3 + \frac{\Delta_t}{R}p(t) \tag{4}$$

So, with a little relabelling, the discrete oscillator takes the form,

$$y_{i+1} = \alpha y_i - \beta y_i^3 + \gamma p(t) \tag{5}$$

And the stochastic resonance problem is now to maximise the kurtosis of the response, by varying the parameter vector $\underline{\theta} = (\alpha, \beta, \gamma)$.

3. SADE

The standard Differential Evolution (DE) algorithm of reference [9] attempts to transform a randomly generated initial population of parameter vectors into an optimal solution through repeated cycles of evolutionary operations, in this case: *mutation*, *crossover* and *selection*. In order to assess the suitability of a certain solution, a fitness function is needed; the kurtosis of the oscillator response is used here. Figure 2 shows a schematic for the DE procedure for evolving between populations. The following process is repeated with each vector within the current population being taken as a *target vector*; each of these vectors has an associated fitness (kurtosis). Each target vector is pitted against a *trial vector* in a competition which results in the vector with highest fitness advancing to the next generation.

The mutation procedure used in basic DE runs as follows. Two vectors A and B are randomly chosen from the current population to form a vector differential $A - B$. A *mutated* vector is then obtained by adding this differential, multiplied by a scaling factor F, to a further randomly chosen vector C to give the overall expression for the mutated vector: $C + F(A - B)$. The scaling factor, F, is often found have an optimal value between 0.4 and 1.0.

The *trial vector* is the child of two vectors: the target vector and the mutated vector, and is obtained via a crossover process; in this work, uniform crossover is used. Uniform crossover decides which of the two parent vectors contributes to each chromosome of the trial vector by a series of $D - 1$ binomial experiments. Each experiment is mediated by a crossover parameter C_r (where $0 \leq C_r \leq 1$). If a random number generated from the uniform distribution on [0,1] is greater than C_r, the trial vector takes its parameter from the target vector, otherwise the parameter comes from the mutated vector.

This process of evolving through the generations is repeated until the population becomes dominated by only a few high fitness solutions, any of which would be suitable. Like the vast majority of optimisation algorithms, convergence to the global maximum is not guaranteed; however, one of the benefits of the evolutionary approach is that it more resistant to finding a local maximum.

A potential weakness of the standard implementation of the DE algorithm as described above is that it requires the prior specification of a number of *hyperparameters* (parameters which need to be specified before the algorithm can run). Apart from the population size, maximum number of iterations etc., the algorithm needs *a priori* specification of the scaling factor F and crossover probability C_r. The values used are not guaranteed to work as well in all situations and an algorithm which establishes 'optimum' values for these parameters during the course of the evolution is clearly desirable. Such an algorithm is available in the form of the Self-Adaptive Differential Evolution (SADE) algorithm [10, 11]; the description and implementation of the algorithm here largely follows [11].

The development of the SADE algorithm begins with the observation that Storn and Price, the originators of DE, arrived at five possible strategies for the mutation operation [12]:

(i) *rand1*: $M = A + F(B - C)$

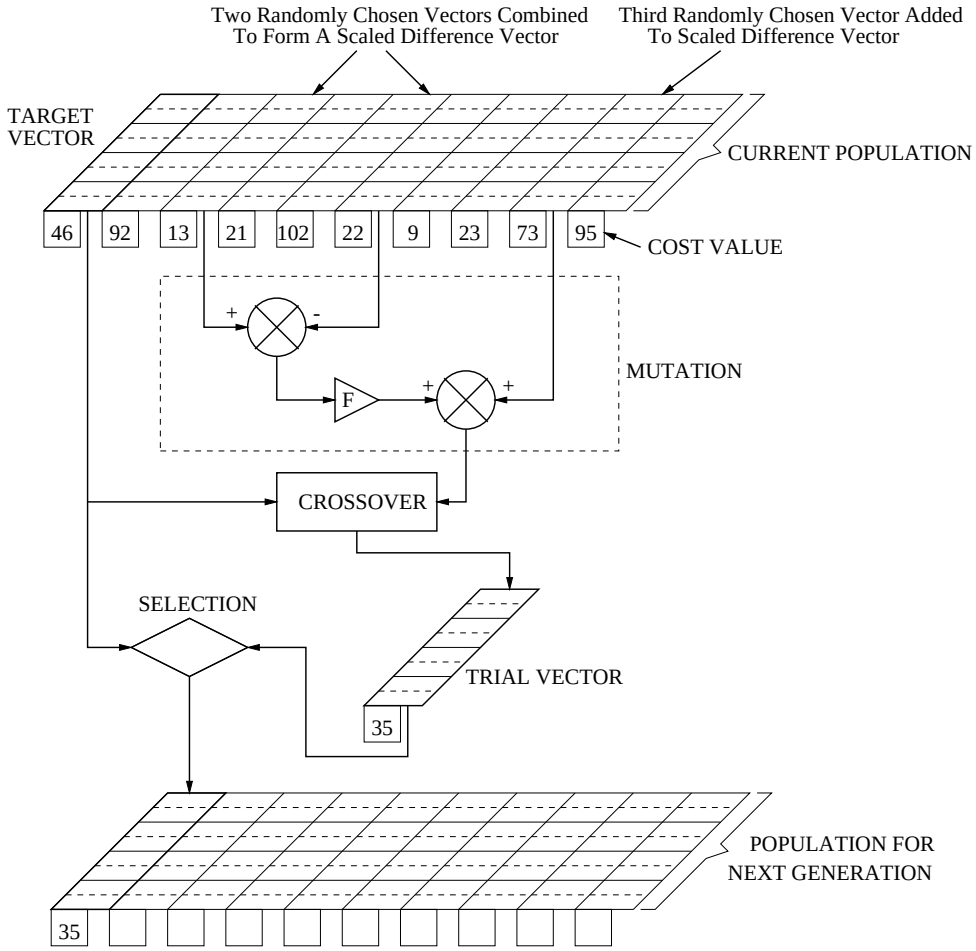

Figure 2. Schematic for the standard DE algorithm.

(ii) *best1*: $M = X^* + F(B - C)$
(iii) *current-to-best*: $M = T + F(X^* - T) + F(B - C)$
(iv) *best2*: $M = X^* + F(A - B) + F(C - D)$
(v) *rand2*: $M = A + F(B - C) + F(D - E)$

where T is the current trial vector, X^* is the vector with (currently) best cost and (A, B, C, D, E) are randomly-chosen vectors in the population distinct from T. F is a standard (positive) scaling factor. The SADE algorithm also uses multiple variants of the mutation algorithm as above; however these are restricted to the following four:

(i) *rand1*
(ii) *current-to-best2*: $M = T + F(X^* - T) + F(A - B) + F(C - D)$
(iii) *rand2*
(iv) *current-to-rand*: $M = T + K(A - T) + F(B - C)$

In the strategy *current-to-rand*, K is defined as a coefficient of combination and would generally be assumed in the range [-0.5,1.5]; however, in the implementation of [11] used here, the prescription $K = F$ is used to essentially restrict the number of tunable parameters. The SADE algorithm uses the standard crossover approach, except that at least one crossover is

forced in each operation on the vectors. If mutation moves a parameter outside its allowed (predefined) bounds, it is pinned to the boundary. Selection is performed exactly as in DE; if the trial vector has higher (or equal) fitness to the target, it replaces the target in the next generation.

The adaption strategy must now be defined. First, a set of probabilities are defined: $\{p_1, p_2, p_3, p_4\}$, which are the probabilities that a given mutation strategy will be used in forming a trial vector. These probabilities are initialised to be all equal to 0.25. When a trial vector is formed during SADE, a roulette wheel selection is used to choose the mutation strategy on the basis of the probabilities (initially, all equal). At the end of a given generation, the numbers of trial vectors successfully surviving to the next generation from each strategy are recorded as: $\{s_1, s_2, s_3, s_4\}$; the numbers of trial vectors from each strategy which are discarded are recorded as: $\{d_1, d_2, d_3, d_4\}$. At the beginning of a SADE run, the survival and discard numbers are established over the first generations, this interval is called the *learning period* (and is another example of a hyperparameter). At the end of the learning period, the strategy probabilities are updated by,

$$p_i = \frac{s_i}{s_i + d_i} \tag{6}$$

After the learning period, the probabilities are updated every generation, but using survival and discard numbers established over a moving window of the last N_L generations. The algorithm thus adapts the preferred mutation strategies. SADE also incorporates adaption or variation on the hyperparameters F and C_r. The scaling factor F mediates the convergence speed of the algorithm, with large values being appropriate to global search early in a run and small values being consistent with local search later in the run. The implementation of SADE used here largely follows [10] and differs only in one major aspect, concerning the adaption of F. Adaption of the parameter C_r is based on accumulated experience of the successful values for the parameter over the run. It is assumed that the crossover probability for a trial is normally distributed about a mean $\overline{C}_r$ with standard deviation 0.1. At initiation, the parameter C_r is set to 0.5 to give equal likelihood of each parent contributing a chromosome. The crossover probabilities are then held fixed for each population index for a certain number of generations and then resampled. In a rather similar manner to the adaption of the strategy probabilities, the C_r values for trial vectors successfully passing to the next generation are recorded over a certain greater number of generations and their mean value is adopted as the next $\overline{C}_r$. The record of successful trials is cleared at this point in order to avoid long-term memory effects. The version of the algorithm here adapts F in essentially the same manner as C_r but uses the Gaussian $N(0.5, 0.3)$ for the initial distribution. At this point, the reader might legitimately argue that SADE has simply replaced one set of hyperparameters (F, C_r) with another (duration of the learning period etc.). In fact, because DE and SADE are heuristic algorithms, there is no analytical counter to this argument. However, the transition to SADE is justified by the fact that the algorithm appears to be very robust with respect to the new hyperparameters.

4. Data

The optimisation algorithm will be benchmarked here on the two synthetic data sets examined in [6]. The first set represents a single impulse submerged in noise, while the second considers an impulse train in noise.

4.1. Single Impulse

The first data set was generated by summing a single impulse response function and Gaussian noise, as follows,

$$p(t) = s(t) + n(t) = Ae^{-D(t-T_i)} \sin(2\pi f_0[t - T_i]) + N(0, \sigma) \tag{7}$$

The parameters used were as follows: $A = 0.15$, $D = 12$, $T_i = 1.0$, $f_0 = 16$ Hz and $\sigma = 0.07$. A sampling frequency of 500 Hz was adopted and 1000 samples were generated; this placed the impulse in the centre of the record. As the resampling ratio R is absorbed in the discrete parameters, its value is not important, although it was set at 200 in [6].

4.2. Impulse Train

This is a more realistic simulation, as real machine faults can manifest as impulse trains embedded in noise. For example, faults in bearings can manifest in several ways due to different causes, such as fatigue, wear, poor installation, improper lubrication and occasionally manufacturing faults. As mentioned in the introduction, the period of the impulse train can give indications of the type of fault e.g. different characteristic frequencies arise: ball pass frequency of the outer race, ball pass frequency of the inner race, fundamental train frequency, ball spin frequency.

In order to simulate an impulse train, the individual impulses were set as the impulse responses of an SDOF system with a resonance frequency of 5000 Hz and a damping ratio of 5%; their amplitude was set as $A = 0.30$. The noise was defined as a Gaussian white noise sequence with zero-mean and standard deviation $\sigma = 0.07$. In [6], the resampling factor was set at $R = 20000$. The signal was sampled at a frequency of 96000 Hz and a second of data was collected.

5. Results

The optimisation problem was now investigated on the two data sets. The initial ranges for the SADE algorithm are important so they were initially set here by consideration of equations (3) and (4) which suggest that α will be a number a little greater than unity, and β and γ will be small positive numbers. There is usually a small trial and error exercise with SADE, where one tunes the initial ranges by making a small number of runs and expanding the ranges until no final estimates are pinned at either end of the range. The tuning phase led to inital ranges of $[1.0, 1.4]$, $[0.001, 2]$, $[0.001, 1]$ for α, β and γ respectively. In all runs, the population size was set at 30 individuals, and the number of generations was set at 1000.

The results for the single impulse case will be presented first. Because SADE is a stochastic algorithm based on a randomly-generated initial population, it is usual to make several runs and save the results from the best one. In this case, 10 runs of the algorithm were made and the optimum kurtosis obtained was 13.0. This is not as high as the result in [6]; however, one should bear in mind, that a different oscillator model is being used here. The fitness curves over the 10 runs are shown in Figure 3, with the maximum fitness for each run in red, and the average fitness in blue. The average fitnesses in each run are much lower than the maximum fitnesses because many of the candidate solutions generated in each generation cross the oscillator potential barrier at the origin and are automatically assigned low fitness. The input and ouput signals for the optimum oscillator are given in Figure 4; the impulse is very clearly amplified in the output.

For the second case study - the impulse train, only 5 runs on the SADE algorithm were carried out because the signal comprised 96000 points. The best solution arrived at a kurtosis of 19.3, which is only slightly worse than the 19.5 reported in [6]. The fitness curves for the runs are given in Figure 5, and the input and output signals for the optimised oscillator are given in Figure 6.

The results are very encouraging. During the multiple SADE runs carried out, there were still instances of parameter estimates being pinned to the ends of the initial ranges, so it is possible that improvements can still be made by expanding the initial ranges.

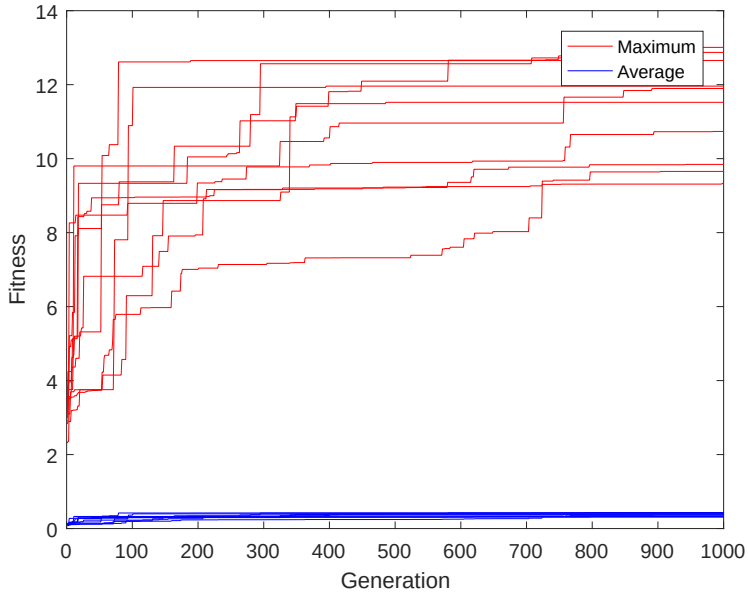

Figure 3. Average and maximum fitness values for 10 SADE runs on example 1: a single impulse in noise.

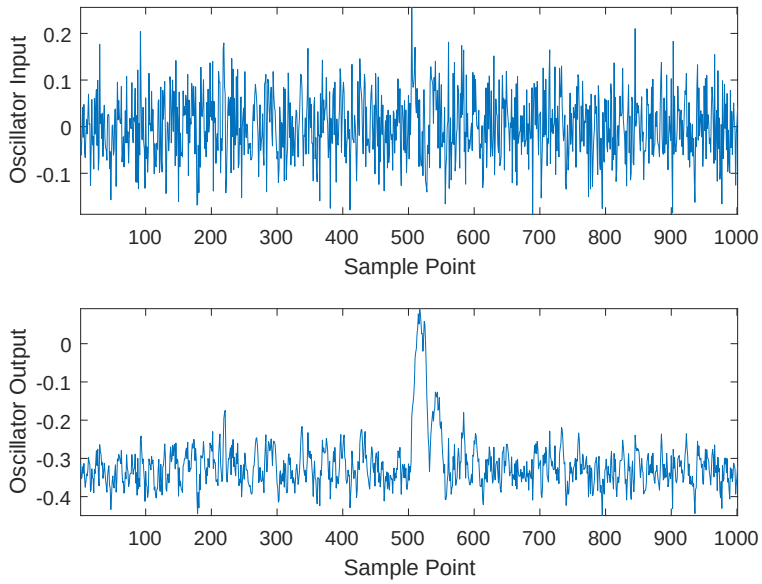

Figure 4. Input and output to SR oscillator for example 1: input kurtosis = 2.83; output kurtosis = 13.0.

6. Conclusions

The paper presents two new ideas in terms of using stochastic resonance for the detection of impulses in background noise. The first is simply to use an evolutionary optimisation scheme

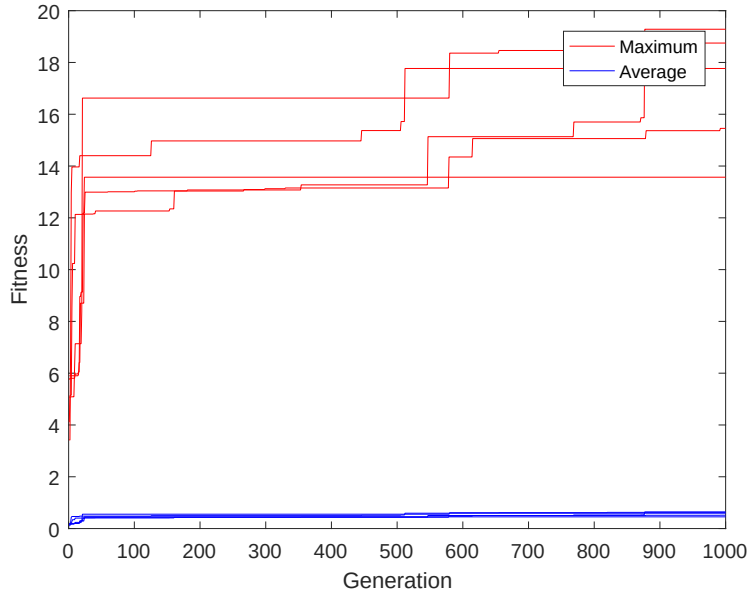

Figure 5. Average and maximum fitness values for 5 SADE runs on example 2: a train of impulses in noise.

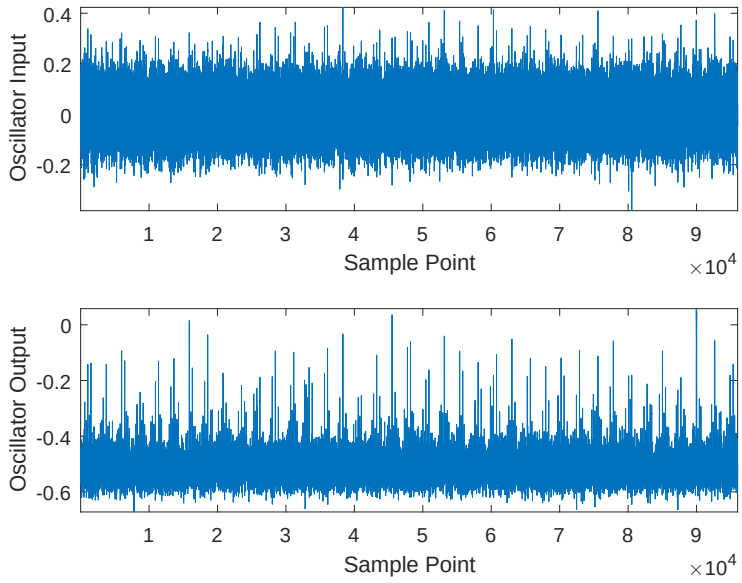

Figure 6. Input and output to SR oscillator for example 2: input kurtosis = 3.38; output kurtosis = 19.3.

to set the oscillator parameters. Because optimisation uses forward prediction through the oscillator model in order to evaluate the objective, it is very slow if one uses the 'traditional' continuous-time representation of the oscillator, so this has been replaced with a discrete-time

variant here. The initial results on simulated data are promising, and the next stage will be to evaluate the approach on experimental data. It is anticipated that some small adaptations to the SADE hyperparameters may occur then, so the simulated data may be re-evaluated. The ultimate aim is to freeze all parameters of the algorithm so that it provides a universal solution, independent of problem.

This paper has been concerned with the process of *feature extraction* for condition monitoring; the next paper in the (short) series will consider *anomaly* or *novelty detection*.

Acknowledgments
The authors would like to acknowledge the UK Engineering and Physical Sciences Research Council (EPSRC) for funding this research through the AIMaREM project (EPSRC grant reference number: EP/N018427/1); they would also like to thank Dr Graeme Manson of the Dynamics Research Group, for providing the original code for the SADE algorithm.

References

[1] Benzi R, Sutera A and Vulpiani A 1981 *J. Phys. A* **10** L453.
[2] McDonnell M D and Abbot D 2009 *PloS Comp. Bio.* **5**.
[3] Li J, Chen X and He Z 2013 *J. Sound Vib.* **332** 5999.
[4] Li J, Chen X and He Z 2013 *Mech. Sys. Sig. Proc.* **36** 240.
[5] Lu S, He Q and Kong F 2014 *Mech. Sys. Sig. Proc.* **45** 488.
[6] Marchesiello S, Fasana A, Garibaldi L 2015 *Proc. Int. Conf. on Structural Engineering Dynamics (Lagos, Portugal)*.
[7] Zhou P, Lu S, Liu F, Liu Y, Li and Zhao J 2017 *J. Sound Vib.* **391** 194.
[8] Tan J, Chen X, Wang J, Chen H, Cao H, Zi Y and He Z 2009 *Mech. Sys. Sig. Proc.* **23** 811
[9] Price K and Storn R 1997 *J. Glob. Opt.* **11** 341.
[10] Qin A K and Suganthan P N 2005 *Proc. IEEE Congress on Evolutionary Computation (Edinburgh, Scotland)* 1785.
[11] Huang V L, Qin A K and Suganthan 2006 *Proc. IEEE Congress on Evolutionary Computation (Vancouver, Canada)* 17.
[12] Storn R and Price K 2009 http://www.icsi.berkeley.edu/ storn/code.html: (Accessed 27th October 2009).,